Các Cấu Trúc Điều Khiển

(Control Statements)

Trịnh Tấn Đạt
Khoa CNTT - Đại Học Sài Gòn
Email: trinhtandat@sgu.edu.vn
Website: https://sites.google.com/site/ttdat88/

Nội dung

- Câu lệnh if … else
- Vòng lặp for
- Vòng lặp while , do … while
- Câu lệnh break, continue
- Các ví dụ minh họa
- Bài tập

Cấu Trúc Điều Khiển

❖ Nhắc lại lệnh và khối lệnh (**statement and block statement**)

- Một biểu thức trở thành một lệnh khi nó được kết thúc bởi dấu ";", được gọi là lệnh đơn.
- Các dấu { và } dùng để nhóm các khai báo và lệnh đơn → lệnh ghép hay khối lệnh.
- Về mặt cú pháp, khối lệnh tương đương một lệnh đơn.
- Các khối lệnh được dùng trong định nghĩa hàm, dùng với các cấu trúc điều khiển chương trình như: if, else, while, for,...

Cấu Trúc Điều Khiển

❖ Cấu Trúc Điều Khiển (**control statement**)

- Cấu trúc điều khiển **xác định thứ tự** các lệnh được thực thi.
- Có 2 loại cấu trúc điều khiển thường gặp:
  - Cấu trúc ***"chọn"*** (selection statements - ***if, switch) biểu diễn các*** "quyết định"
  - Cấu trúc ***"lặp"*** (iteration statements or loop ***- for, while) cho phép lặp lại*** nhiều lần việc thực thi các lệnh.
- Ví dụ
  - **Nếu** a> b **thì** a là số lớn nhất, **ngược lại nếu** b>a **thì** b là số lớn nhất, **ngược lại** a bằng b
  - In ra màn hình **1000 lần** chuỗi "Hello World".

Câu lệnh if … (thiếu)

❖ **Câu lệnh if …**

- Cú pháp:

```
if (expr)
    câu lệnh;
```

```
if (expr)
{
    khối lệnh;
}
```

- Lưu ý:
  - expr là **biểu thức logic bắt buộc** đặt trong dấu ngoặc ( ) và KHÔNG có dấu chấm phẩy ở cuối.
  - Nếu phần lệnh bên trong if chỉ là **lệnh đơn** thì KHÔNG cần dấu { }
  - Nếu phần lệnh bên trong if là **khối lệnh** thì cần dấu { }. Thông thường, nếu chỉ là lệnh đơn cũng nên để trong dấu { } để nhận biết rằng đó là câu lệnh bên trong if.
- **Ý nghĩa:** Nếu biểu thức expr là **đúng** (true, 1, khác 0) thì thực thi câu lệnh hoặc khối lệnh bên trong if ngược lại thì bỏ qua.

Ví dụ

Lệnh đơn

```
int a = 0;
if (a== 0)
      cout <<" a bang 0";
```

```
int a = 0;
if (a== 0)
{
      cout <<" a bang 0";
}
```

```
int a = 0;
if (a==0)
{
      cout <<" a bang 0";
      a++;
      int b = 10;
}
```

Lệnh kép

Lưu ý: phép so sánh == khác phép gán =

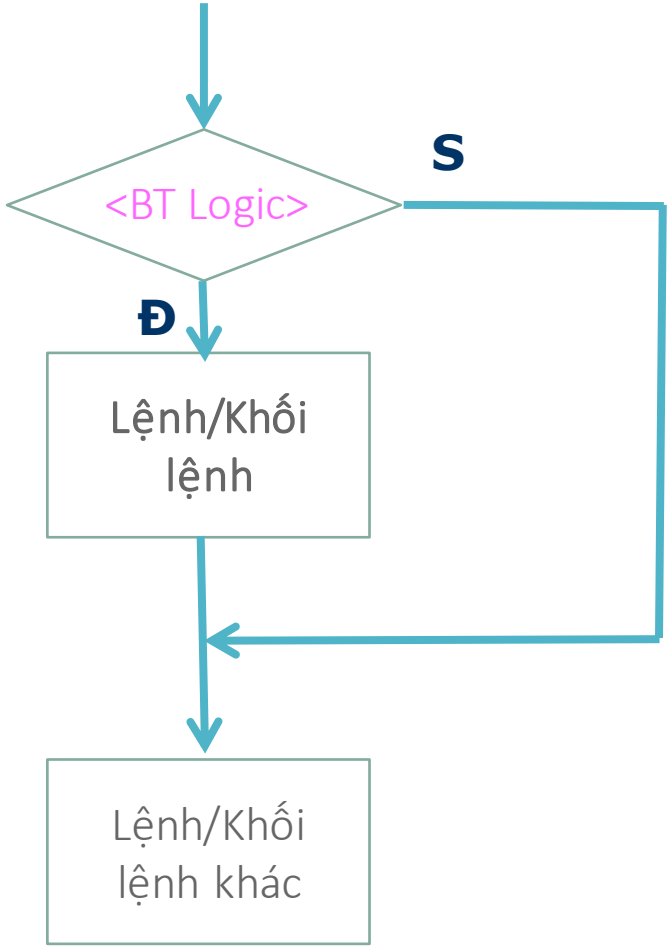
```
int a = 0;
if (a=0)
      cout <<" a bang 0";
```

SAI

Câu lệnh if …

❖ **Câu lệnh if …**

- Flow chart:



Ví dụ: if …

```
#include <iostream>
using namespace std;
int main()
{
    int a;
    cout <<"nhap vao a :";
    cin >> a;
    if (a>0)
        cout << "a la so duong"<<endl;
    if (a<0)
    {
        cout<< "a la so am" <<endl;
    }
    if (a==0)
        cout << "a bang 0";

    return 0;
}
```

Ví dụ: if …

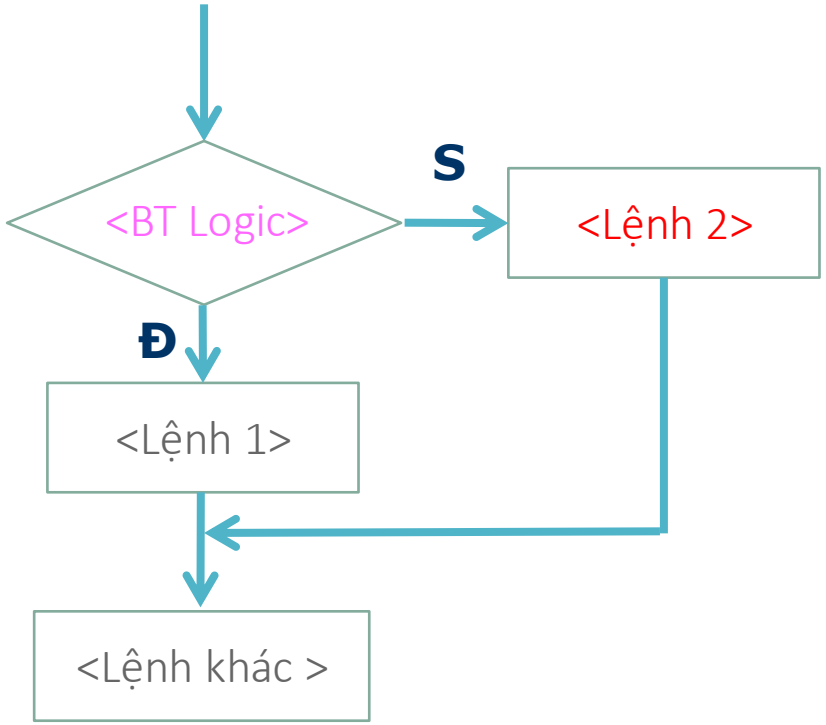
```
#include <iostream>
using namespace std;
int main()
{
    int a,b,c;
    int m;
    cout <<"nhap vao a, b, c :";
    cin >> a >> b >> c;
    m = a;
    if (m < b)
        m = b;
    if (m < c)
        m = c;
    cout <<"MAX la:" << m;

    return 0;
}
```

Câu lệnh if … else … (đủ)

❖ **Câu lệnh if … else …**

- Cú pháp:

```
if (expr)
{
    lệnh1;
}
else
{
    lệnh2;
}
```



- Ý nghĩa: Nếu biểu thúc expr là **đúng** (true, 1, khác 0) thì thực thi câu lệnh 1 ngược lại nếu expr là **sai** thì thực hiện câu lệnh2.

Ví dụ

```
int a; cin >> a;
if (a==0)
      cout <<" a bang 0";
else
      cout << " a khac 0";
```

Lệnh đơn

```
int a; cin >> a;
if (a == 0)
{
      cout <<" a bang 0";
      a++;
}
else
{
      cout << " a khac 0";
      a--;
}
```

Khối lệnh

Ví dụ: if … else …

```
#include <iostream>
using namespace std;
int main()
{
   unsigned int a;
   cout <<"nhap vao so nguyen duong a:"; // a > 0
   cin >> a ;
   if (a%2==0)  //chia cho 2 phần dư là 0
      cout << "a la so chan" <<endl;
   else
      cout << "a la so le" <<endl;

   return 0;
}
```

Lưu ý: if … ; if … else

1. Câu lệnh if và câu lệnh if… else là một câu lệnh đơn.

```
{
    if (a == 0)
        cout <<"a bang 0";
}

{
    if (a == 0)
    {
        cout<<"a bang 0";
        a = 2912;
    }
    else
        cout<"a khac 0";
}
```

Lưu ý: if … ; if … else

2. Câu lệnh if có thể lồng vào nhau và else sẽ tương ứng với if gần nó nhất.

```
if (a != 0)
   if (b > 0)
      cout<<"a != 0 va b > 0";
else
   cout<< "a != 0 va b <= 0";

// Cách viết khác (canh lề) dễ phân biệt hơn
if (a !=0)
{
   if (b > 0)
       cout<<"a != 0 va b > 0";
   else
       cout<< "a != 0 va b <= 0";
}
```

Lưu ý: if … ; if … else

3. Nên dùng else để loại trừ trường hợp.

```
if (delta < 0)
   cout<<"PT vo nghiem";
if (delta == 0)
   cout<<"PT co nghiem kep";
if (delta > 0)
   cout<<"PT co 2 nghiem";

// Cách viết khác rõ ràng hơn
if (delta < 0)
   cout<<"PT vo nghiem";
else  // delta >= 0
{  if (delta == 0)
      cout<<"PT co nghiem kep";
   else
      cout<<"PT co 2 nghiem";
}
```

Lưu ý: if … ; if … else

4. Mệnh đề **if… else if… else …** trong C++

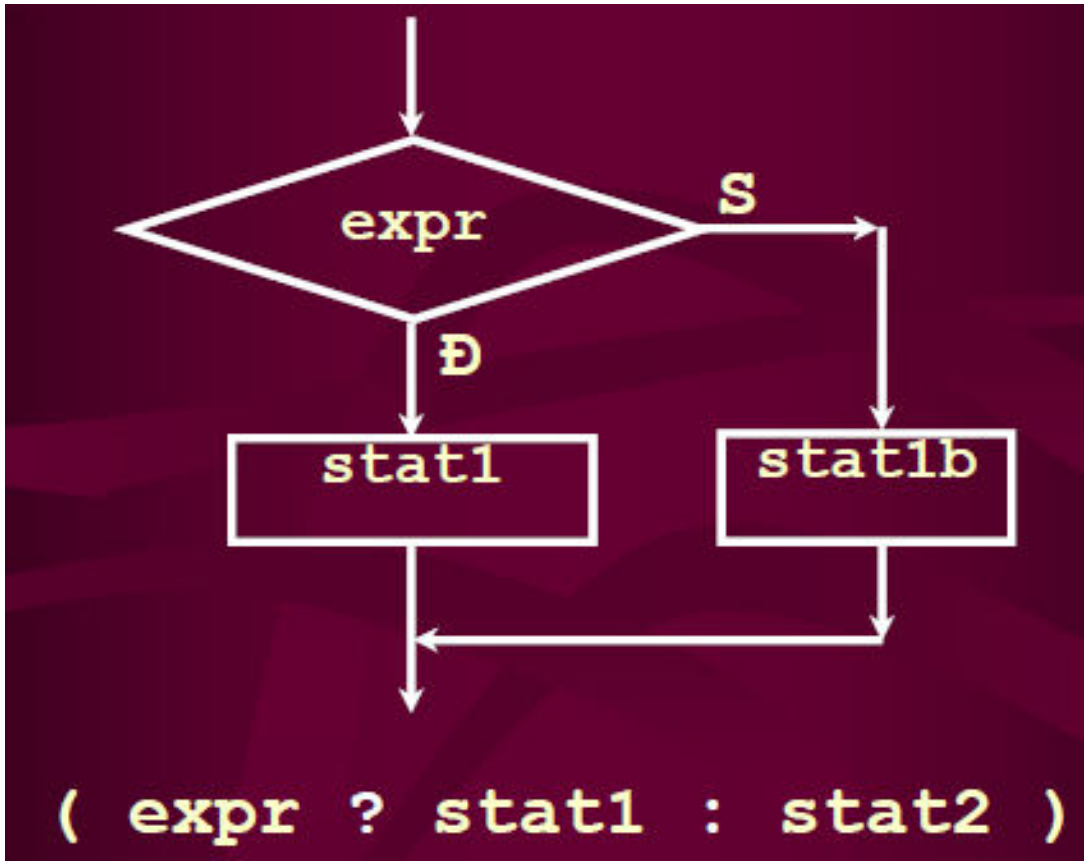
```
if (condition1)
{
        // khối lệnh này được thực thi nếu condition1 = true
}
else if (condition2)
{
        // khối lệnh này được thực thi nếu condition1 = false và condition2 = true
}
else
{
        // khối lệnh này được thực thi nếu nếu tất cả những điều kiện trên = false
}
```

Ví dụ:

```
#include <iostream>
using namespace std;
int main()
{
   int num;
   cout<<"Nhap 1 so de kiem tra thang diem:";
   cin>>num;
   if (num < 0 || num >= 100) {
      cout<<"Ban nen nhap so tu 1 den 100";
   }
   else if(num > 0 && num < 50) {
      cout<<"Fail";
   }
   else if ( num >= 50 && num < 60) {
      cout<<"D Grade";
   }
   else if ( num >= 60 && num < 70) {
      cout<<"C Grade";
   }
   else if ( num >= 70 && num < 80) {
      cout<<"B Grade";
   }
   else if ( num >= 80 && num < 90) {
      cout<<"A Grade";
   }
   else if ( num >= 90 && num <= 100) {
      cout<<"A+ Grade";
   }
    return 0;
}
```

Lưu ý: if … ; if … else

5. Mối liên hệ giữa if .. else .. và toán tử điều kiện (toán tử ba ngôi)



Tìm max{a, b} ?

```
m = a>b ? a : b;
```

Tìm |a| ?

```
m = a>0 ? a : -a;
```

Lưu ý: if … ; if … else

6. Không được thêm ; sau điều kiện của if.

Why???

```
2. a la so chan
3. a la so chan


...Program finished with exit code 0
```

```
#include <iostream>
using namespace std;
int main()
{
    int a = 11;
    if (a%2==0)
        cout << "1. a la so chan" <<endl;

    if (a%2==0);
        cout << "2. a la so chan" <<endl;

    if (a%2==0)
    {
    };
    cout << "3. a la so chan" <<endl;

    return 0;
}
```

Cấu trúc switch … case …

❖ **Cấu trúc switch … case …** (dạng thiếu) : switch → chương trình chọn lựa một trong nhiều phương án khác nhau tùy thuộc **kết quả của biểu thức so sánh bằng**.

- Cú pháp:

```
switch (<Biến/BT>)
{
        case <GT1>:<L1>;   break;
        case <GT2>:<L2>;   break;
        …
        case <GTN>:<LN>;  break;
}
```

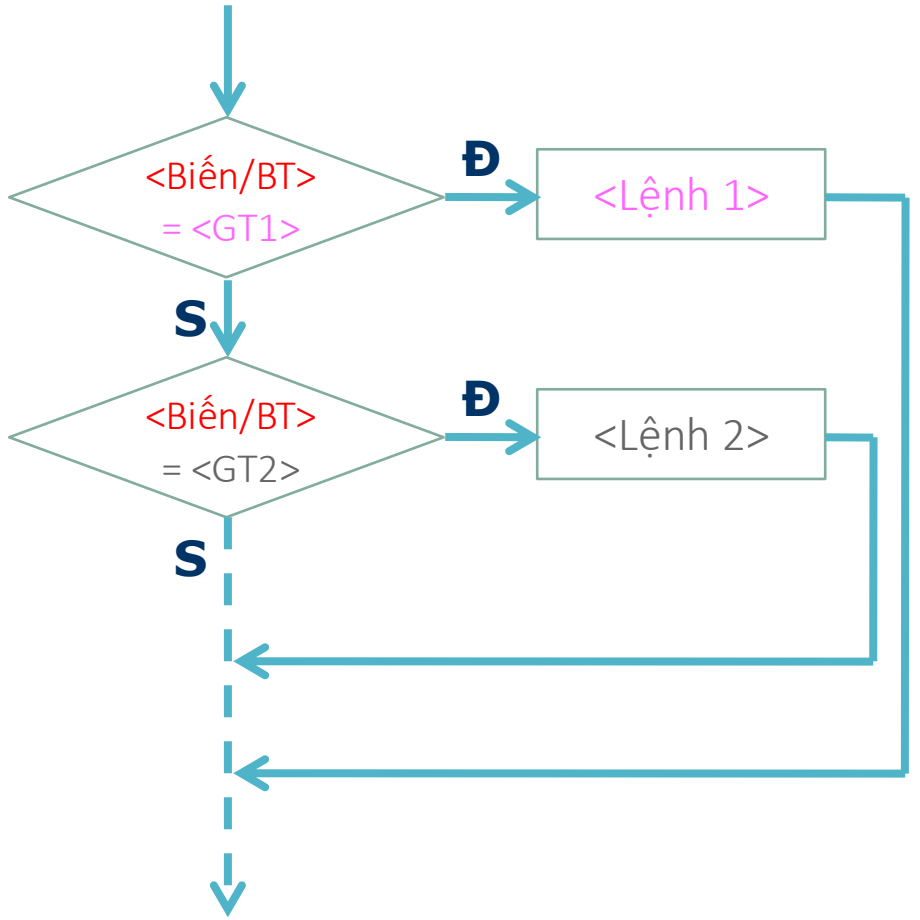
Trong đó:
- <Biến/BT> là biến/biểu thức cho giá trị rời rạc.
- Các giá trị **<GT1> , … ,<GTN>:** là các hằng và không được trùng nhau.
- Lệnh **<L1>, … <LN>**: là các lệnh đơn hoặc khối lệnh { }
- Câu lệnh **break;** : kết thúc thực thi và thoát khỏi switch

Ý nghĩa :
-Nếu **<Biến/BT>** trùng với giá trị **<GTK>** **nào đó, thì thực hiện câu lệnh <LK>** rồi kết thúc . (với K từ 1 đến N)
- Nếu **<Biến/BT> KHÔNG** trùng với giá trị **<GT> nào, thì không thực hiện gì.**

Cấu trúc switch … case … (thiếu)

- Flow chart

```
switch (<Biến/BT>)
{
        case  <GT1>:<L1>;   break;
        case  <GT2>:<L2>;   break;
        …
        case  <GTN>:<LN>;  break;
}
```



Ví dụ

```
#include <iostream>
using namespace std;
int main()
{
    int a;
    cout <<"nhap vao a:";
    cin >>a;
    switch (a)
     {
            case 1 : cout<<"Mot"; break;
            case 2 : cout<<"Hai"; break;
            case 3 : cout<<"Ba"; break;
       }
    return 0;
}
```

Ví dụ

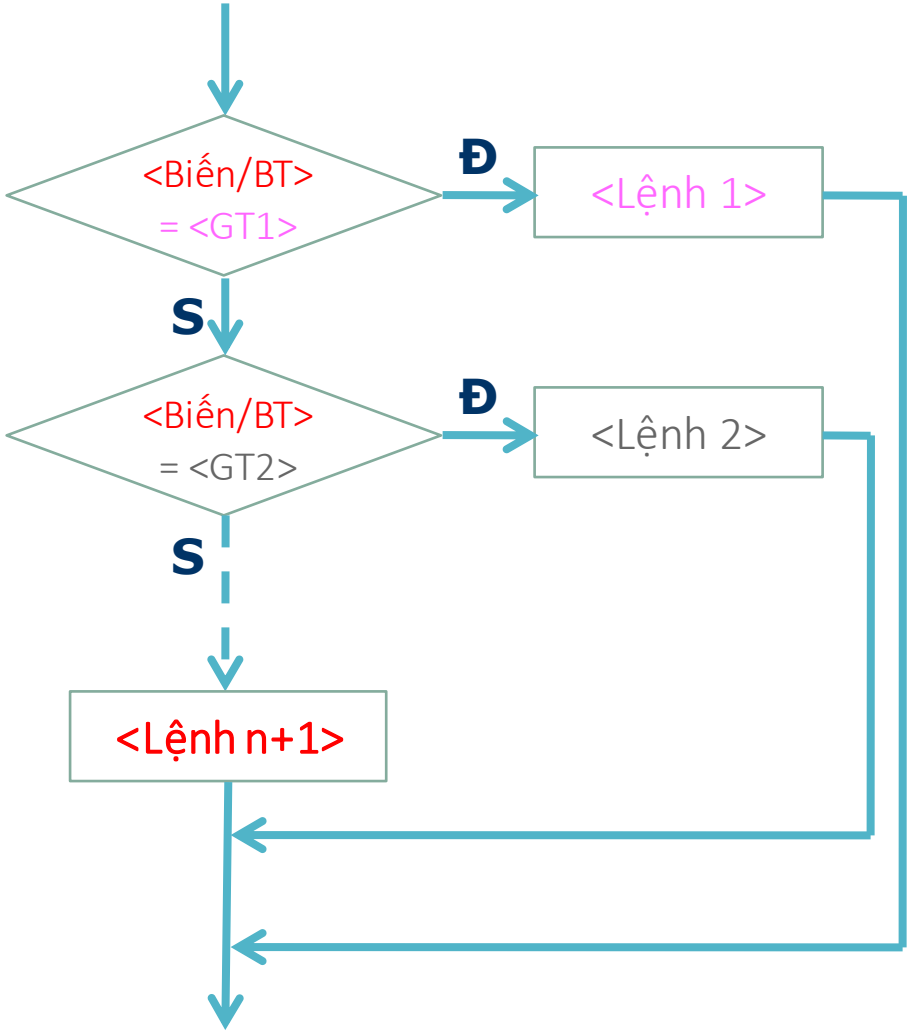
```
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout <<"nhap vao thang trong nam a:"; // thang 1 den thang 12
   cin >>a;
   switch (a)
   {
            case 1: case 2: case 3:
               cout<<"Quy I";  break;
            case 4: case 5: case 6:
               cout<<"Quy II";  break;
           case 7: case 8: case 9:
              cout<<"Quy III";  break;
           case 10: case 11: case 12:
              cout<<"Quy IV";  break;
      }
   return 0;
}
```

Cấu trúc switch … case …

❖ **Cấu trúc switch … case …** (dạng đầy đủ): có từ khóa **default**

- Cú pháp:

```
switch (<Biến/BT>)
{
        case <GT1>:<L1>;   break;
        case <GT2>:<L2>;   break;
        …
        case <GTN>:<LN>;  break;
        default:
                <LN+1>;
}
```



Ý nghĩa :
- Nếu **<Biến/BT> KHÔNG** trùng với tất cả kỳ giá trị **<GT> trong case,** thì sẽ thực hiện câu lệnh **<LN+1> tại default.**

Ví dụ:

```
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout <<"nhap vao a:";
   cin >>a;
   switch (a)
    {
            case 1 : cout<<"Mot"; break;
            case 2 : cout<<"Hai"; break;
            case 3 : cout<<"Ba"; break;
            default : cout<<"Khong the chuyen doi";
       }
   return 0;
}
```

Ví dụ

```
#include <iostream>
using namespace std;
int main()
{
   int a, b;

   cout <<"nhap vao a va b:";
   cin >> a >> b;
   char tt;
   cout <<"nhap vao phep toan:";
   cin >> tt; // phep toan + hoac -
   switch (tt)
    {
            case '+' : cout<<a<<"+"<<b<<"="<<a+b;
                    break;
            case '-' : cout<<a<<"-"<<b<<"="<<a-b;
                    break;
            default : cout<<"Phep toan ko hop le";
       }
   return 0;
}
```

Lưu ý: switch … case …

- Câu lệnh switch là một **câu lệnh đơn** và có thể **lồng nhau**.

```
{
   switch (a)
   {
      case 1 : cout<<"Mot"; break;
      case 2 :
               { switch (b)
                  {
                  case 1 : cout<<"A"; break;
                  case 2 : cout<<"B"; break;
                  }
               } break;
      case 3  : cout<<"Ba"; break;
      default : cout<<"Khong doc duoc";
   }
}
```

Lưu ý: switch … case …

- switch sẽ nhảy đến case tương ứng và **thực hiện đến khi nào gặp break** hoặc cuối switch sẽ kết thúc.

```
nhap vao a:1
Mot

...Program finished with exit code 0
```

WHY???

```
nhap vao a:2
HaiBaKhong the chuyen doi

...Program finished with exit code 0
```

```
#include <iostream>
using namespace std;
int main()
{
   int a;
   cout <<"nhap vao a:";
   cin >>a;
   switch (a)
   {
            case 1 : cout<<"Mot"; break;
            case 2 : cout<<"Hai";
            case 3 : cout<<"Ba";
            default : cout<<"Khong the chuyen doi";
      }
   return 0;
}
```

Lưu ý:

- Kinh nghiệm khi nào sử dụng if … else khi nào sử dụng switch … case …

Câu lệnh if

```
if (a == 1)
   cout<<"Mot";
if (a == 2)
   cout<<"Hai";
if (a == 3)
   cout<<"Ba";
if (a == 4)
   cout<<"Bon";
//...
```

Câu lệnh switch

```
switch (a)
{
   case 1:   cout<<"Mot";
      break;
   case 2:   cout<<"Hai";
      break;
   case 3:   cout<<"Ba";
      break;
   case 4:   cout<<"Bon";
      break;
}
```

Lưu ý:

- Kinh nghiệm khi nào sử dụng if … else khi nào sử dụng switch … case …

Câu lệnh if

```
if (a == 3.14)
   cout<<"OK";
if (a < 10)
    cout<<"OK";
if (a == 1)
    cout<<"OK";
if (a == 2 || a == 3)
    cout<<"OK";
```

Câu lệnh switch

```
switch (a)
{
case 3.14: // ko dc
case <10:  // ko dc
case 1: cout<<"OK";
     break;
case 2:
case 3: cout<<"OK";
     break;
}
```

Câu hỏi

- Hãy cho cho biết, có lỗi nào trong chương trình sau ?

```
#include <iostream>
using namespace std;
int main()
{
   int x;
   cin>>x;
   if( x > 0)
       cout<<" x duong !";
   else (x < 0)
       cout<<" x am !";
   else
       cout<<" x khong am khong duong !";
   return 0;
}
```

Cấu trúc lặp

- Cấu trúc lặp (**iteration statement, loop**): dùng để lặp lại một hành động trong khi còn thỏa một điều kiện nào đó.

Ví dụ

Viết chương trình xuất các số từ 1 đến 10

=> Sử dụng 10 câu lệnh printf hoặc cout

Viết chương trình xuất các số từ 1 đến 1000

=> Sử dụng 1000 câu lệnh printf hoặc cout

- Có 03 lệnh lặp: for, while, do… while

Vòng lặp for

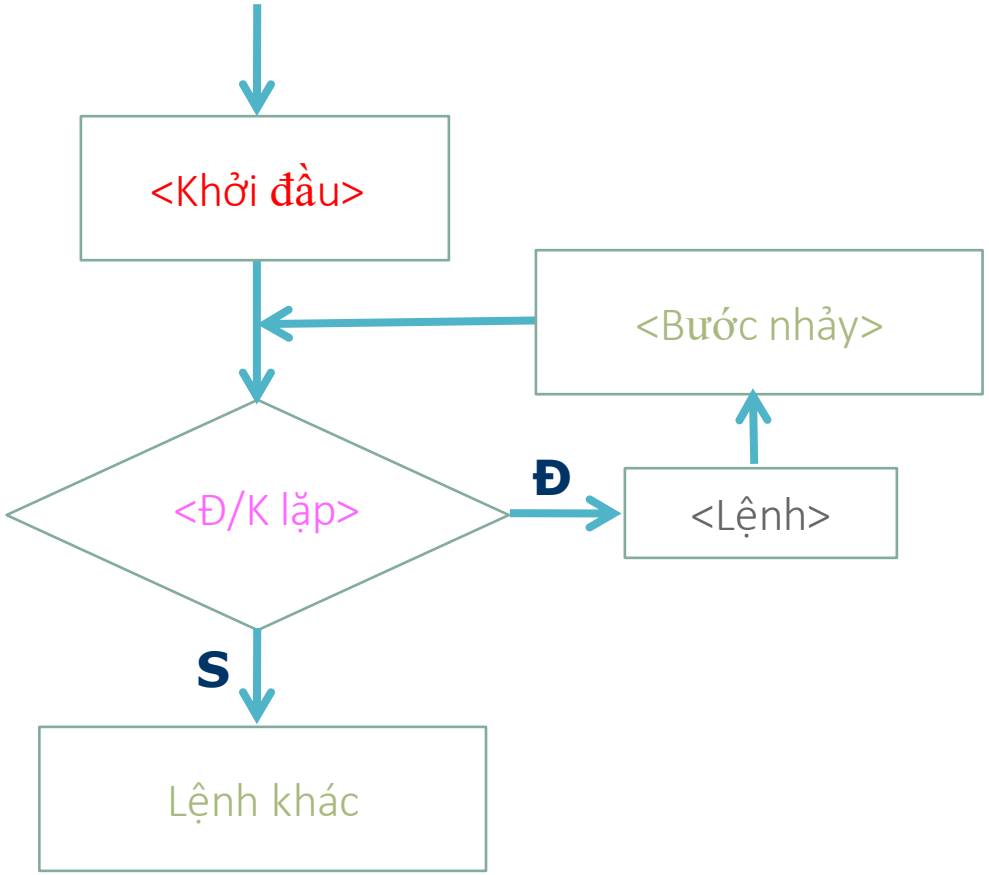
- Cú pháp:

```
for (<Khởi đầu>; <Đ/K lặp>; <Bước nhảy>)
{
        <Lệnh>;  // lệnh đơn hoặc khối lệnh
}
```

- Lưu ý: Khởi đầu>, <Đ/K lặp>, <Bước nhảy> là biểu thức C bất kỳ có chức năng riêng
  - <Đ/K lặp> là **biểu thức logic**. Nếu biểu thức trong điều kiện lặp là **ĐÚNG** thì tiếp tục thực hiện lệnh trong vòng lặp. Nếu **SAI** sẽ thoát khỏi vòng lặp
  - <Khởi đầu> thường là giá trị đầu tiên cho biến đếm .
  - <Bước nhảy> thường là thay đổi giá trị biến điếm theo bước nhảy
  - 3 thành phần <Khởi đầu>;<Đ/K lặp>;<Bước nhảy> phân tách nhau bởi dấu chấm phẩy ';' và đặc trong dấu ngoặc đơn ()
  - KHÔNG có dấu chấm phẩy phía sau for

Vòng lặp for

- Flow chart và ý nghĩa



Khi phát biểu for thực thi:

B1. Gán trị ban đầu: **dem = gtbd** trong <Khởi đầu>

B2. Kiểm tra biểu thức trong <Đ/K lặp>

B3. Nếu biểu thức là TRUE (≠0), thì sang B4.

Nếu biểu thức là FALSE (=0), thì sang B6.

B4. Các phát biểu (<Lệnh>) trong phần thân của for thực thi.

B5. Thay đổi giá trị biến đếm theo <Bước nhảy>

Quay trở về B2.

B6. Kết thúc for, lệnh khác sau for tiếp tục thực thi

Ví dụ

- In giá trị từ 1 đến 1000
- In các số chẵn từ 1 đến 1000
- In các số lẻ từ 1 đến 1000

```
#include <iostream>
using namespace std;
int main()
{
   int i;
   for(i=1;i<=1000;i++)
   {
        cout<< i << "  ";
   }
   cout<<endl;

   for(i=2;i<=1000;i=i+2)
      cout<< i << "  ";
   cout<<endl;

   for(i=1;i<=1000;i=i+2)
      cout<< i << "  ";
   cout<<endl;
   return 0;
}
```

Ví dụ

- Tính tổng

$S = 1+2+3+ \ldots + n$

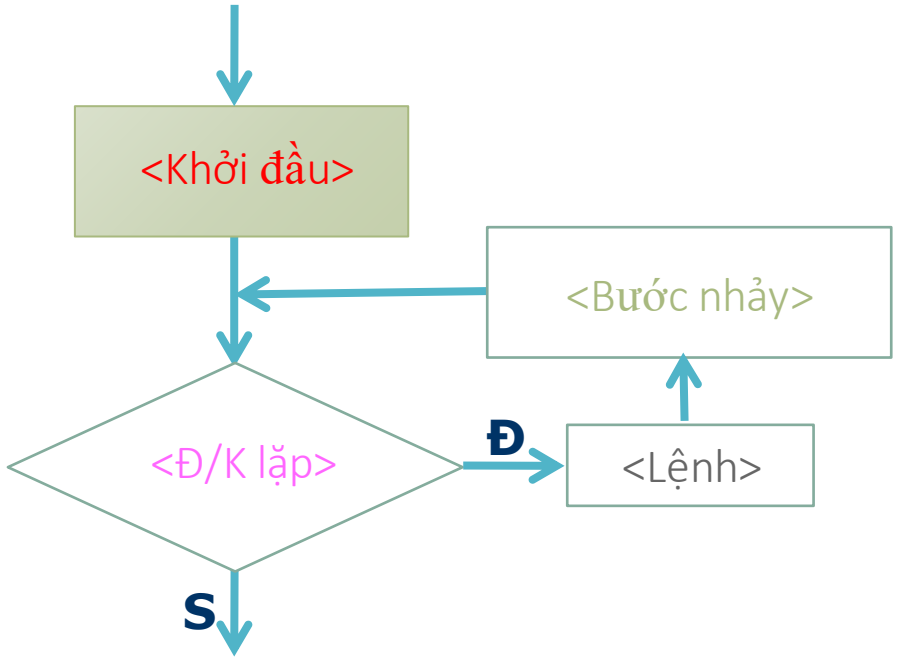
n là số nguyên nhập từ bàn phím

```
#include <iostream>
using namespace std;
int main()
{
   int i,n;
   cout<<"nhap vao n:";
   cin>>n;
   int tong = 0; //dieu gi xay ra neu khong gan bang 0 truoc
   for(i=1;i<=n;i++)
      tong = tong+i; // hay viet tuong duong tong+=i;
   cout <<"ket qua:" <<tong<<endl;
   return 0;
}
```

Lưu ý: for

1. Trong câu lệnh for, có thể sẽ không có phần <Khởi đầu>

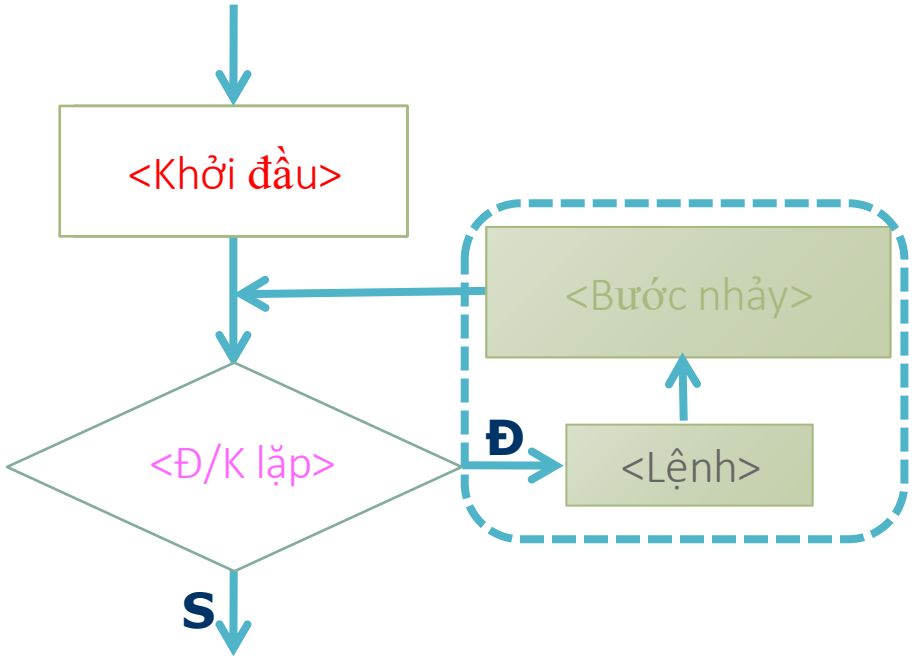
```
int i;
for (i = 0; i < 10; i++)
    cout<<i <<" ";

// hoặc cách khác
int i = 0; // gán trước rồi for
for (; i < 10; i++)
   cout<<i <<" ";
```



Lưu ý: for

2. Trong câu lệnh for, có thể sẽ không có phần <Bước nhảy>

```
int i;
for (i = 0; i < 10; i++)
    cout<<i <<" ";

//cách khác
for (i = 0; i < 10; )
{
   cout<<i <<" ";
   i++;
}
```



Lưu ý: for

3. Trong câu lệnh for, có thể sẽ không có phần <Đ/K lặp>

```
int i;
for (i = 0; i < 10; i++)
   cout<<i <<" ";

for (i = 0; ; i++)
   cout<<i <<" ";

//cách khác
for (i = 0; ; i++)
{
   if (i >= 10)
      break; // khi gặp break lặp tức thoát khỏi for
   cout<<i <<" "; // lệnh này trong for và ngoài if
}
```

Lệnh break: sẽ giới thiệu ở phần slide phía sau

Lưu ý: for

4. Không được thêm ; ngay sau lệnh lệnh for.

=> Tương đương câu lệnh rỗng.

```
for (i = 0; i < 10; i++);
{
   cout<<"i"<< endl;
}
//tương đương
for (i = 0; i < 10; i++)
{
};
{
    cout<<"i"<<   endl;
}
```

Câu hỏi: sau câu lệnh này i bằng mấy?

Lưu ý: for

5. Các thành phần <Khởi đầu>, <Đ/K lặp>, <Bước nhảy> cách nhau bằng dấu ;

Nếu có nhiều thành phần trong mỗi phần thì được cách nhau bằng dấu ,

```
int i,j;
for (int i = 1, j = 2; i + j < 10; i++, j += 2)
      cout<< i + j;
```

Lưu ý: for

6. Câu lệnh FOR là một câu lệnh đơn và có thể lồng nhau.

Câu hỏi: ý nghĩa đoạn code là gì?

```
if (n < 10 && m < 10)
{
   for (int i = 1; i < n; i++)
   {
      for (int j = 1; j <= m; j++)
      {
         cout<<i<<"*"<<j<<"="<<i*j;
         cout<<endl;
      }
      cout<<endl;
   }
}
```

Ví dụ:

- Vẽ hình chữ nhật đặc bằng các dấu '*'

```
#include <iostream>
#include <stdio.h>
using namespace std;
int main()
{
   int i, j, idai, irong;
      printf("Nhap vao chieu dai: ");
      scanf("%d", &idai);
      printf("Nhap vao chieu rong: ");
      scanf("%d", &irong);
      for (i = 1; i <= irong; i++)
      {
            for (j = 1; j <= idai; j++) //in mot hang voi chieu dai dau *
            {
                  printf("*");
            }
            printf("\n");  //xuong dong khi in xong 1 hang
      }
   return 0;
}
```

Vòng lặp while

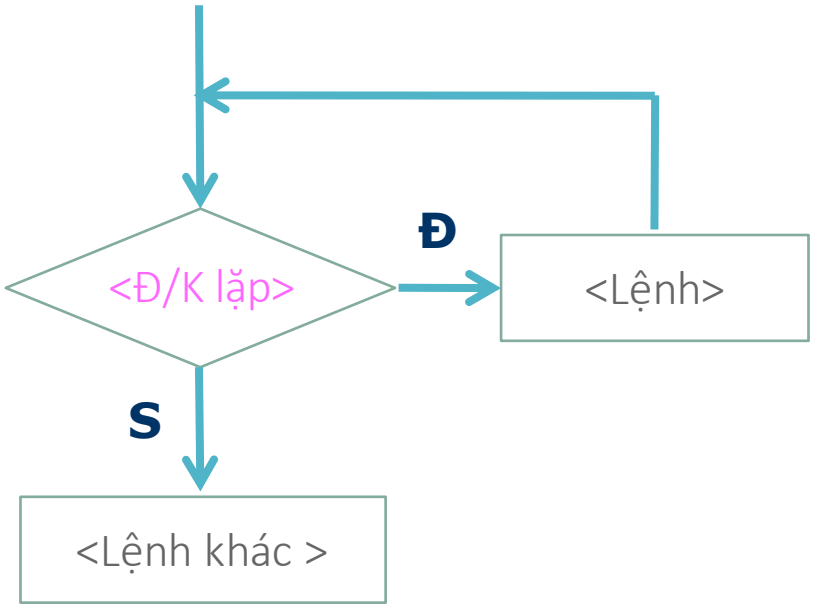
- Cú pháp:

```
while (<Đ/K lặp>)
{
        <Lệnh>; // lệnh đơn hoặc khối lệnh
}
```

- Lưu ý:
  - <Đ/K lặp> là **biểu thức logic**. Nếu biểu thức trong điều kiện lặp là **ĐÚNG** thì tiếp tục thực hiện lệnh trong vòng lặp. Nếu **SAI** sẽ thoát khỏi vòng lặp

Vòng lặp while

- Flow chart và ý nghĩa



Khi phát biểu while thực thi:

B1. Kiểm tra biểu thức trong <Đ/K lặp>

B2. Nếu biểu thức là TRUE (≠0), thì sang B3.

Nếu biểu thức là FALSE (=0), thì sang B4.

B3. Các phát biểu (<Lệnh>) trong phần thân của while được thực thi. Sau đó, quay lại bước B1.

B4. Kết thúc while, lệnh khác sau while tiếp tục thực thi

Vòng lặp while

- Ví dụ: 03 đoạn code sau đây đều in ra giá trị từ 0 đến 9

```
int i = 0;
while (i < 10)
{
    cout<<"i"<<endl;
    i++;
}
```

```
for (int i = 0; i < 10; i++)
    cout<<"i"<<endl;
```

```
int i = 0;
for (; i < 10; )
{
    cout<<"i"<<endl;
    i++;
}
```

Ví dụ

- Nhập vào một giá trị n là số nguyên dương. Nếu nhập sai yêu cầu nhập lại

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cin>>n;
    while(n<=0)
    {
        cout<<"nhap vao n";
        cin>>n;
    }
    cout<<"n la:" <<n<<endl;
    return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
    int n= -1;
    while(n<=0)
    {
        cout<<"nhap vao n:";
        cin>>n;
    }
    cout<<"n la:" <<n<<endl;
    return 0;
}
```

Ví dụ

- Nhập vào số nguyên dương n. Điếm số lượng số nguyên dương nhỏ hơn n và chia hết cho 3.

Câu hỏi: in ra các số thỏa điều kiện trên

```
#include <iostream>
using namespace std;
int main()
{
   int i, dem, n = -1;
   while(n<=0)
   {
      cout<<"nhap vao n:";
      cin>>n;
   }
   i = 1, dem = 0;   // i chay tu 1 den n-1,
   while(i<n)
   {
        if(i%3==0)
              dem++;

        i++;
   }
   cout<<"Co "<<dem<<" so nho hon "<<n<<" va chia het cho 3"<<endl;
   return 0;
}
```

Lưu ý: while

1. Cần xác định các yếu tố:

- Điều kiện lặp = **điều kiện "làm"**

  ≠ điều kiện "dừng".

  - Làm những gì ?
  - Yếu tố làm biểu thức logic của while thay đổi ?

```
while(i<n)
  {
        if(i%3==0)
              dem++;

        i++;  // biến i thay đổi
  }
```

Lưu ý: while

2. Câu lệnh while là một câu lệnh đơn và có thể lồng nhau.

```
if (n < 10 && m < 20)
{
   while (n >= 1)
   {
      while (m >= 1)
      {
         cout<<m;
         m--;
      }
      n--;
   }
}
```

Lưu ý: while

3. Câu lệnh while có thể không thực hiện lần nào do điều kiện lặp ngay từ lần đầu đã không thỏa.

```
int main()
{
    int n = 1;
    while (n > 10)
    {
        cout<<n<<endl;
        n--;
    }
    //…
    return 0;
}
```

Lưu ý: while

4. Câu lệnh while có thể bị lặp vô tận (loop)

- Tính tổng S = 1+1/2+1/3+1/4+…+1/n , n là số nguyên dương nhập từ bàn phím.

```
#include <iostream>
using namespace std;
int main()
{
   int i, n;
   cout<<"nhap vao n:";
   cin>>n;
   double S=0;
   i=1;
   while(i<=n)
   {
      S = S + 1.0/i;  // S= S+1/i; ????
   }
   cout<<"Tong la:"<<S<<endl;
   return 0;
}
```

```
#include <iostream>
using namespace std;
int main()
{
   int i, n;
   cout<<"nhap vao n:";
   cin>>n;
   double S=0;
   i=1;
   while(i<=n)
   {
      S = S + 1.0/i;  // S= S+1/i; ????
       i++;
   }
   cout<<"Tong la:"<<S<<endl;
   return 0;
}
```

Lưu ý: while

5. Không được thêm ; ngay sau lệnh lệnh while.

```
int n = 0;
while (n < 10);
{
    cout<<n<<endl;
    n++;
}
// tương đương
while (n < 10)
{
};
{
    cout<<n<<endl;
    n++;
}
```

Vòng lặp do … while

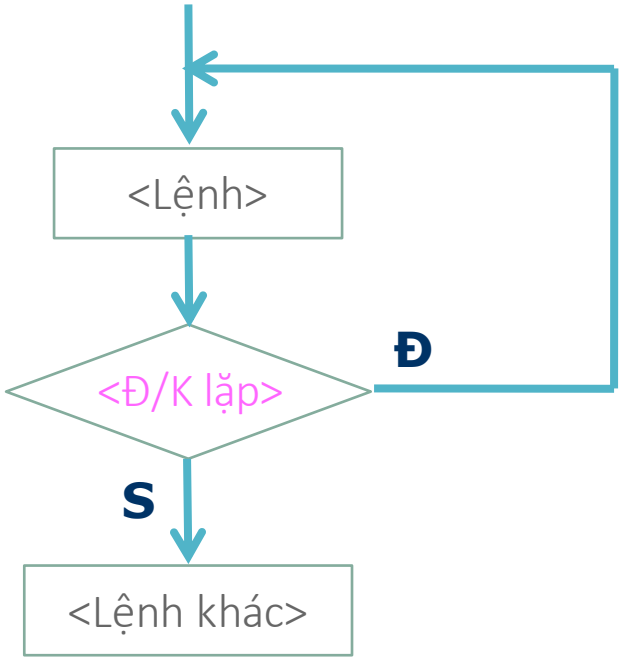
- Cú pháp:

```
do
{
    <Lệnh>;
}
while (<Đ/K lặp>);
```

- Lưu ý:
  - <Đ/K lặp> là **biểu thức logic**. Nếu biểu thức trong điều kiện lặp là **ĐÚNG** thì tiếp tục thực hiện lệnh trong vòng lặp. Nếu **SAI** sẽ thoát khỏi vòng lặp.
  - **Có dấy chấm phẩy ; cuối while**

Vòng lặp while

- Flow chart và ý nghĩa



Khi phát biểu do… while thực thi:

B1. Các lệnh trong thân của do…while thực thi

B2. Kiểm tra biểu thức trong <Đ/K lặp>

B3. Nếu biểu thức là TRUE (≠0), thì sang **B1**.

Nếu biểu thức là FALSE (=0), thì kết thúc do… while, thực hiện các lệnh khác.

⇒ **Như vậy,** Câu lệnh do… while sẽ được thực hiện ít nhất 1 lần do điều kiện lặp được kiểm tra ở cuối. .

Vòng lặp do … while

- Ví dụ:

```
int i = 0;
do
{
    cout<<i<<endl;
    i++;
}
while (i < 10);

// tương dương cách viết khác
int i = 0;
cout<<i<<endl;
i++;
for (; i < 10; )
{
    cout<<i<<endl;
    i++;
}
```

Ví dụ

- Nhập vào từng số nguyên dương n, nhập đến đâu tính tổng các số vừa nhập đến đó. Nhập vào số âm để kết thúc quá trình nhập.

```
#include <iostream>
using namespace std;
int main()
{
   int n, tong =0;
   do
   {
       cout<<"nhap vao n:";
            cin>>n;
            if(n>0)
            {
               tong += n;
               cout<<"tong la:" <<tong<<endl;
            }
   }while(n>=0);
   return 0;
}
```

Lưu ý: do … while

1. Câu lệnh do… while là một câu lệnh đơn và có thể lồng nhau

```
int a = 1, b;
do
{
   b = 1;
   do
   {
      cout<<a+b<<endl;
      b = b + 2;
   }
   while (b < 20);
   a++;
}
while (a < 20);
```

Lưu ý: do … while

2. Câu lệnh do… while có thể bị lặp vô tận (loop)

```
//…
    int n = 1;
    do
    {
        cout<<n;
        n--;
    }
    while (n < 10);

    n = 1;
    do
        cout<<n;
    while (n < 10);
//…
```

Lưu ý: do … while

3. while có thể không thực hiện lần nào.

   do… while sẽ được thực hiện ít nhất 1 lần.

```
int n = 100;
while (n < 10)
{
    cout<<n;
}
// su khac biet …
do
{
    cin>> n;
}
while (n > 10);
```

Lệnh break

- Lệnh **break** **kết thúc cấu trúc lặp gần nhất** mà nó xuất hiện trong đó, không cần biết kết quả của biểu thức logic expr (ĐK lặp)
- Được dùng trong trường hợp **thoát khỏi vòng lặp** mà **không dùng đến điều kiện dừng.**
- Lệnh break thường xuất hiện cùng với phát biểu if.
- break còn được dùng để thoát khỏi switch, nếu các nhóm lệnh (case i) không được kết thúc bằng break thì máy có thể đi từ case i sang case i+1.

Lệnh break

- Ví dụ:

```
int i;
for (i = 1; i < 10; i++)
{
    if (i % 2 == 0)
        break;
    cout<<i<<endl;
}
```

```
// kết quả in ra 1
// khi i=2 thỏa Đk if, gặp break sẽ thoát ngay khỏi for
```

Lệnh break

- Ví dụ: Kiểm tra số nguyên dương n có phải là số nguyên tố hay không? (n nhập từ bàn phím)

```
#include <iostream>
using namespace std;
int main()
{   int n;
   cout<<"nhap vao so nguyen duong n:";
   cin>>n;
   if (n<2)
   {
      cout<<n <<" khong la so nguyen to";
      return 0;
   }
   int i, flag = 0; // flag == 0 la so nguyen to
   for(i=2;i<n;i++)
   {
      if(n%i==0) // n chia het cho i
      {
         flag = 1;  // flag ==1 khong la so ngueyn to
         break;
      }
   }
   if (flag)  // tuong duong viết  flag == 1
      cout<<n <<" khong la so nguyen to";
   else
      cout<<n <<"  la so nguyen to";
   return 0;
}
```

// DK này tốt hơn???
for(i=2;i<sqrt(n)+1;i++)0

Lệnh continue

- Lệnh continue dùng để bắt đầu một lần lặp mới của cấu trúc lặp gần nhất mà nó xuất hiện trong đó.
  - Khi gặp continue, vòng lặp sẽ bỏ qua phần code bên dưới continue, vòng lặp sẽ tiếp tục bước lặp tiếp theo (quay về bước nhảy (nếu có) và kiểm tra biểu thức logic (ĐK lặp)).
- Cụ thể:
  - Khi gặp continue bên trong phát biểu for, máy sẽ chuyển tới bước 5 trong "sự hoạt động của for" (slide 63).
  - Khi gặp continue bên trong phát biểu while hoặc do-while, máy sẽ chuyển tới bước kiểm tra biểu thức (bước 1 trong while, bước 2 trong do-while).
- Phát biểu continue thường xuất hiện cùng với phát biểu if.

Lệnh continue

- Ví dụ:

```
for (i = 0; i < 10; i++)
{
    if (i % 2 == 0)
        continue;
    cout<<i<<endl;
}
```

Kết quả: In các số lẻ từ 0 đến 10 (các số chẵn bị bỏ qua không in ra màn hình)

Lệnh continue

- Ví dụ: Tính tổng các bội số của 5 nhỏ hơn số nguyên dương n. (n nhập từ bàn phím)

```
#include <iostream>
#include <math.h>
using namespace std;
int main()
{
   int n,i;
   int tong =0;
   cout<<"nhap vao so nguyen duong n:";
   cin>>n;
   for(i=1;i<n;i++)
   {
      if(i%5!=0) //bo qua so khong chia het cho 5
          continue;
      tong +=i;
   }
   cout<<"tong la:"<<tong<<endl;
   return 0;
}
```

Các ví dụ minh họa

- Viết chương trình nhập vào 2 số nguyên a, b. Nếu a lớn hơn b thì hoán đổi giá trị a và b, ngược lại không hoán đổi. In ra giá trị a, b.

```
#include <stdio.h>
int main()
{
	int ia, ib, itam;
	printf("Nhap vao so a: ");
	scanf("%d", &ia);
	printf("Nhap vao so b: ");
	scanf("%d", &ib);
	if (ia>ib)
	{
		itam = ia; //hoan vi a va b
		ia = ib;
		ib = itam;
	}
	printf("%d, %d\n", ia, ib);
	return 0;
}
```

Các ví dụ minh họa

- Viết chương trình nhập vào số nguyên n. Tính tổng các giá trị lẻ từ 0 đến n.

```
#include <iostream>
using namespace std;
int main()
{
	int i, n, tong;
	cout<<"Nhap vao so n: ";
	cin>>n;
	tong = 0;
	for(i = 0; i<=n; i++)
	{
		if (i % 2 != 0) //neu i la so le
			tong = tong + i; //hoac tong += i;
	}
	cout<<"tong la:"<<< tong;
	return 0;
}
```

Các ví dụ minh họa

Tính tổng $S(x,n) = -x + \frac{x^2}{2!} - \frac{x^3}{3!} + ... + (-1)^n \frac{x^n}{n!}$

```
#include <stdio.h>
int main()
{
        int n;     scanf("%d",&n);
        float x;  scanf("%f",&x);
       float tong = 0.0;     int tuso = 1;        int mauso = 1;        int dau = 1;
        int i = 1;
        while(i<=n)
        {
                dau = dau*(-1);
                tuso = tuso*x;
                mauso = mauso*i;
                tong = tong + ((float)tuso/mauso)*(dau);
                i++;
        }
        printf("tong = %.3f", tong);
        return 0;
}
```

Các ví dụ minh họa

THCSLT3-03

Tính tổng S = 1.2 + 2.3.4 +3.4.5.6+...+ n(n+1)...(2n)

```
#include <stdio.h>
int main()
{
	int n;
	scanf("%d",&n);
	int tong = 0;
	for(int i = 1; i <= n ; i++)
	{
		int tich = 1;
		for (int j = i; j<=2*i; j++)
			tich = tich*j;
		tong = tong + tich;
	}

	printf("tong = %d\n", tong);
	return 0;
}
```

Bài Tập: If else, switch case

1. Nhập vào a và b và giải phương trình bậc nhất ax+b =0.
2. Nhập vào a , b và c . Giải pt bậc hai $ax^2+bx+c = 0$.
3. Nhập vào một chữ cái (kiểu ký tự). Nếu là in thương thì chuyển sang IN HOA và ngược lại

Ví dụ: nhập và a thì chuyển sang A. Nhập vào B thì chuyển sang b

4. Nhập vào 4 số nguyên a,b,c,d. Tìm số lớn nhất và nhỏ nhất trong 4 số vừa nhập.
5. Nhập 4 số nguyên a, b, c và d. Hãy sắp xếp giá trị của 4 số nguyên này theo thứ tự tăng dần.

Bài Tập: If else, switch case

6. Nhập vào tháng và năm. Cho biết tháng đó có bao nhiêu ngày.
7. Nhập độ dài 3 cạnh 1 tam giác. Kiểm tra đó có phải là tam giác không và là tam giác gì?

Bài Tập: Vòng lặp

Nhập vào số nguyên dương n và tính các tổng sau:

1. Tính $S = 1^3 + 2^3 + 3^3 + \ldots\ldots + N^3$
2. Tính $S = 1^2 + 2^2 + 3^2 + \ldots\ldots + N^2$
3. Tính $S = 1 + 1/2 + 1/3 + \ldots + 1/n$
4. Tính $S = 1/(1x2) + 1/(2x3) + 1/(3x4) + \ldots\ldots + 1/(n \times (n+1))$
5. Tính $S = 1 + 1.2 + 1.2.3 + \ldots + 1.2.3\ldots n$
6. Tính $S = 1 + x + x^2 + \ldots\ldots + x^n$

Bài Tập: Vòng lặp

7. In ra tất cả các số nguyên tố nhỏ hơn n (với n là số nguyên dương nhập từ bàn phím)

Ví dụ: n = 10, in ra: 2 3 5 7

8. Nhập vào số nguyên dương n và số thực x

Tính tổng $S(x,n) = x + \frac{x^2}{1+2} + \frac{x^3}{1+2+3} + \ldots + \frac{x^n}{1+2+3+\ldots+n}$

9. Nhập vào số nguyên dương n và số thực x

Tính tổng $S(x,n) = -x + \frac{x^2}{2!} - \frac{x^3}{3!} + \ldots + (-1)^n \frac{x^n}{n!}$

Bài Tập: Vòng lặp

10. Nhập vào số nguyên dương n . Tính tổng

a) **S = 1*2*3 + 2*3*4 + 4*5*6 + . . .+ n*(n+1)*(n+2)**

b) **S = 2 + (2+4) + (2+4+6) + (2+4+6+8) + … + (2+4+6+8+ … +2n)**

11. Viết chương trình tìm USCLN (ước số chung lớn nhất), BSCNN (bội số chung nho? nhất) của 2 số nguyên dương a và b.

12. Viết chương trình nhập vào N rồi tính giai thừa của N.

Bài Tập: Vòng lặp

13. **THCSLT3-04**

Cho S sẽ được tính theo công thức S=1+3+5+...+(2n+1) với n≥0 và số nguyên M. Hãy tìm giá trị n nhỏ nhất sao cho S > M với M được cho trước. Ví dụ: M = 20 thì n tìm được là 4.

14. **THCSLT3-05**

Viết chương trình hiển thị tất cả các số có 3 chữ số sao cho tổng các chữ số của số đó bằng tích của chúng. Ví dụ: Số 123 là thỏa mãn điều kiện đề bài.

Ví dụ: 1*2*3 = 1+2+3

Bài Tập: Vòng lặp

15. Đếm số lượng chữ số, tính tổng các chữ số của số nguyên dương n

Ví dụ: n = 1234. n có 4 chữ số. Tổng các chữ số là 1+2+3+4 = 10

16. Đếm số lượng chữ số lẻ/ chẵn của số nguyên dương n.

Ví dụ: n = 1234. n có 2 chữ số lẻ là 1, 3.

17. Nhập một số nguyên dương n (n > 0). Hãy cho biết:

    a. Có phải là số đối xứng? Ví dụ: 121, 12321, …

    b. Có phải là số chính phương? Ví dụ: 4, 9, 16, …

    c. Có phải là số nguyên tố? Ví dụ: 2, 3, 5, 7, …

    d. Chữ số lớn nhất và nhỏ nhất?

Bài Tập: Vòng lặp

18. Nhập n. In n số đầu tiên trong dãy Fibonacy.

    a. $a_0 = a_1 = 1$

    b. $a_n = a_{n-1} + a_{n-2}$

19. Nhập vào số nguyên dương n (n >0)

    a) Liệt kê tất cả ước số của số nguyên dương n

    b) Tính tổng các ước số của số nguyên dương n

    c) Đếm số lượng ước số của số nguyên dương n

    d) Tính tổng các ước số chẵn của số nguyên dương n

20. Nhập một số nguyên dương n. Xuất ra số ngược lại. Ví dụ: Nhập 1706 ➔ Xuất 6071